CS420

Introduction to the Theory of Computation

Lecture M3: Module 3 Recap

Tiago Cogumbreiro
Today we will...

- Announcements
- Recap what we learned in CS 420
- Discuss what work/didn't work in CS 420
- Go over sample exercises for mini-test 3
- Course evaluation
Announcements
Mini Test 3
Where: Y02-2330, 2nd University Hall
When: from 5:30pm until 6:45pm
CSM Undergraduate Research Fellowships

Want to do research?

CSM students applying for this Fellowship need to:

- Identify a potential topic of research and a potential research group.
- Demonstrate an excellent academic standing with a minimum grade point average (GPA) of 3.2.
- Commit to working with a research group the equivalent time of 2-3 credits of coursework per semester.
- Agree to present a research poster at the CSM Student Success Showcase on Friday, May 15, 2020.

Deadline: January 15, 2020

forms.umb.edu/csm-opportunities/c/urf
CS 420
Looking back...
Looking back...

- We learned about **regular languages** and their applications (lesson 17)
Looking back...

- We learned about **regular languages** and their applications (lesson 17)
  - regular expressions for validation, text searching
Looking back...

- We learned about **regular languages** and their applications (lesson 17)
  - regular expressions for validation, text searching
  - finite state machines for structuring stateful algorithms
Looking back…

- We learned about **regular languages** and their applications (lesson 17)
  - regular expressions for validation, text searching
  - finite state machines for structuring stateful algorithms
  - multiple abstractions to handle the same concept and solve different problems (DFA/NFA/REGEX)
Looking back…

- We learned context-free languages and their applications (lesson 17)
Looking back...

- We learned **context-free languages** and their applications (lesson 17)
  - context-free grammars for parsing structured text (JSON example/programming languages)
Looking back...

- We learned **context-free languages** and their applications (lesson 17)
  - context-free grammars for parsing structured text (JSON example/programming languages)
  - multiple abstractions to handle the same concept and solve different problems (PDA/CFG)
Looking back...

- We learned about *Turing-recognizable languages*
Looking back...

- We learned about *Turing-recognizable languages*
  - Church-Turing thesis: programs are Turing Machines
Looking back...

- We learned about **Turing-recognizable languages**
  - Church-Turing thesis: programs are Turing Machines
  - Recognizable: implementation answers YES correctly (may loop)
CS 420

Looking back...

- We learned about **Turing-recognizable languages**
  - Church-Turing thesis: programs are Turing Machines
  - Recognizable: implementation answers YES correctly (may loop)
  - Decidable: implementation answers YES/NO correctly (recognizable + terminates)
Looking back...

- We learned about **Turing-recognizable languages**
  - Church-Turing thesis: programs are Turing Machines
  - Recognizable: implementation answers YES correctly (may loop)
  - Decidable: implementation answers YES/NO correctly (recognizable + terminates)
  - Are there decision problems that have no implementation possible?
Looking back...

- We learned about **Turing-recognizable languages**
  - Church-Turing thesis: programs are Turing Machines
  - Recognizable: implementation answers YES correctly (may loop)
  - Decidable: implementation answers YES/NO correctly (recognizable + terminates)
  - Are there decision problems that have no implementation possible?
  - Reducibility: mapping problems into other problems
We learned about **Turing-recognizable languages**
- Church-Turing thesis: programs are Turing Machines
- Recognizable: implementation answers YES correctly (may loop)
- Decidable: implementation answers YES/NO correctly (recognizable + terminates)
- Are there decision problems that have no implementation possible?
- Reducibility: mapping problems into other problems
- Logic programming using a proof assistant
What work/didn't work in CS 420?
CS 420

Do you think using a proof assistant helped you?
CS 420

Do you think we should devote more time learning to use a proof assistant?
Mini Test 3 Primer
Mini Test 3 overview

- 50 points for Sections 4.1 and 4.2 (HW7 + Exercises in Lesson 20)
- around 10 points for Section 5.1
- around 40 points for Section 5.3

- Level 1: 60 points
- Level 2: 25 points
- Level 3: 15 points
Exercise 1 (Level 1)

Know why membership tests fail and succeed; explain **why** certain membership fails.
Exercise 1 (Level 1)

Know why membership tests fail and succeed; explain why certain membership fails.

Let $D$ be the DFA below

\begin{itemize}
  \item Exercise 2.1: Is $\langle D, 0100 \rangle \in A_{DFA}$?
  \item Exercise 2.2: Is $\langle D, 101 \rangle \in A_{DFA}$?
  \item Exercise 2.3: Is $\langle D \rangle \in A_{DFA}$?
  \item Exercise 2.4: Is $\langle D, 101 \rangle \in A_{REX}$?
  \item Exercise 2.5: Is $\langle D \rangle \in E_{DFA}$?
  \item Exercise 2.6: Is $\langle D, D \rangle \in EQ_{DFA}$?
  \item Exercise 2.7: Is $101 \in A_{REX}$?
\end{itemize}
Exercise 2 (Level 1)

Know how to compose decidable algorithms as new decidable algorithms.
Exercise 2 (Level 1)

Know how to compose decidable algorithms as new decidable algorithms.

Give an algorithm that decides $EQ_{REX}$
Exercise 2 (Level 1)

Know how to compose decidable algorithms as new decidable algorithms.

Give an algorithm that decides $EQ_{REX}$

```python
def EQ_REX(R1, R2):
    return EQ_DFA(REX_TO_DFA(R1), REX_TO_DFA(R2))
```

Similar examples: give a decider for

- $A_{NFA}, A_{REX}, A_{PDA}$ (Lesson 17)
- $EQ_{DFA}$ (Lesson 18)
- $EQ_{DFAREX}$ (Exercise 4.2) (or any combination therein)
- $ALL_{DFA}$ (Exercise 4.3)
- $A_{\epsilon_{CFG}}$ (Exercise 4.4)

- $\{\langle R, S \rangle \mid R, S \text{ are regex } \land L(R) \subseteq L(S)\}$ is decidable (Problem 4.13)
- $\{\langle R \rangle \mid R \text{ is regex over } \{0, 1\} \land w \text{ contains 111 } \land w \in L(G)\}$ (Exercise 4.16)
Exercise 3 (Level 1)

Know examples of recognizable, decidable, unrecognizable, undecidable languages.
Exercise 3 (Level 1)

- Know examples of recognizable, decidable, unrecognizable, undecidable languages.

Give an example of a recognizable and undecidable language.
Know examples of recognizable, decidable, unrecognizable, undecidable languages.

Give an example of a recognizable and undecidable language.

**Solution:** $A_{TM}$ is recognizable (in proof of Theorem 4.11, page 202) and undecidable (Theorem 4.11).

Tip: build a table of (co-)recognizable, decidable, undecidable, and (co-)unrecognizable languages

- Think of $A$, $E$, $EQ$ for DFA, CFG, and TM
Exercise 4 (Level 2)

Map-reducible: Use decidability (Theorem 5.22 and Corollary 5.23) and recognizability (Theorem 5.28 and Corollary 5.29) to derive conclusions about the languages we studied ($A, E, EQ + DFA, CFG, TM$).
Exercise 4 (Level 2)

Map-reducible: Use decidability (Theorem 5.22 and Corollary 5.23) and recognizability (Theorem 5.28 and Corollary 5.29) to derive conclusions about the languages we studied (A, E, EQ + DFA, CFG, TM).

Given that $A_{TM} \leq_m HALT_{TM}$, show that $HALT_{TM}$ is undecidable.
Exercise 4 (Level 2)

Map-reducible: Use decidability (Theorem 5.22 and Corollary 5.23) and recognizability (Theorem 5.28 and Corollary 5.29) to derive conclusions about the languages we studied (A, E, EQ + DFA, CFG, TM).

Given that $A_{TM} \leq_m HALT_{TM}$, show that $HALT_{TM}$ is undecidable.

**Proof.** Apply Corollary 5.23 since $A_{TM}$ is undecidable (Theorem 4.11) and $A_{TM} \leq_m HALT_{TM}$ (hypothesis).

More examples

- Show that $HALT_{TM}$ is unrecognizable.
- Show that $HALT_{TM}$ is undecidable. (Exercise 5.24/Lesson 22)
- Show that $A_{TM}$ is recognizable via mapping reducibility. (Lesson 22)
Exercise 5 (level 2)

Relate facts on map-reducible.
Exercise 5 (level 2)

- Exercise 5.6: $\leq_m$ is a transitive relation.
- Exercise 5.22: $A$ is recognizable iff $A \leq_m A_{TM}$.

Let (H1) $A_{CFG} \leq_m A_{TM}$, (H2) $A_{DFA} \leq_m A_{CFG}$, and (H3) $A_{TM}$ is recognizable.

Prove that we can conclude that $A_{DFA}$ is recognizable using map-reducibility.
Exercise 5 (level 2)

- Exercise 5.6: $\leq_m$ is a transitive relation.
- Exercise 5.22: $A$ is recognizable iff $A \leq_m A_{TM}$.

Let (H1) $A_{CFG} \leq_m A_{TM}$, (H2) $A_{DFA} \leq_m A_{CFG}$, and (H3) $A_{TM}$ is recognizable.

Prove that we can conclude that $A_{DFA}$ is recognizable using map-reducibility.

**Proof.**

1. $A_{DFA} \leq_m A_{TM}$ by Exercise 5.6, (H1) $A_{CFG} \leq_m A_{TM}$, (H2) $A_{DFA} \leq_m A_{CFG}$.
2. $A_{DFA}$ is recognizable, by Exercise 5.22, (1) $A_{DFA} \leq_m A_{TM}$, and (H3).
Exercise 6 (Level 2)

Relate facts on map-reducible.
Exercise 6 (Level 2)

Relate facts on map-reducible.

- Lemma R.1: If $A \leq_m B$, then $\overline{A} \leq_m \overline{B}$.
- Lemma R.2: If $A \leq_m \overline{B}$ and $B$ recognizable, then $\overline{A} \leq_m B$.
- Lemma R.3: If $A$ recognizable and $\overline{A} \leq_m B$, then $A \leq_m \overline{B}$.

Let (H1) $B \leq \overline{A}_{TM}$. Show that $\overline{B}$ is recognizable.
Exercise 6 (Level 2)

Relate facts on map-reducible.

- Lemma R.1: If \( A \leq_m B \), then \( \overline{A} \leq_m \overline{B} \).
- Lemma R.2: If \( A \leq_m \overline{B} \) and \( B \) recognizable, then \( \overline{A} \leq_m B \).
- Lemma R.3: If \( A \) recognizable and \( \overline{A} \leq_m B \), then \( A \leq_m \overline{B} \).

Let (H1) \( B \leq \overline{A}_{TM} \). Show that \( \overline{B} \) is recognizable.

**Proof.**

1. \( \overline{B} \leq A_{TM} \), by Lemma R.2, (H1) \( \overline{A}_{TM} \leq B \), and \( A_{TM} \) recognizable (pp 202).
2. \( \overline{B} \) is recognizable, by Exercise 5.22 and (1) \( \overline{B} \leq A_{TM} \).
Exercise 7 (Level 2)

Relate facts on map-reducible.
Exercise 7 (Level 2)

- Relate facts on map-reducible.

Show that $\overline{HALT}_{TM}$ is unrecognizable.
Exercise 7 (Level 2)

Relate facts on map-reducible.

Show that $\overline{HALT}_TM$ is unrecognizable.

Proof.

1. $\overline{A}_TM \leq_m \overline{HALT}_TM$, by Theorem R.1 and $A_{TM} \leq_m HALT_{TM}$ (exercise 5.24)

2. $HALT_{TM}$ is unrecognizable, by Corollary 5.29, $\overline{A}_TM \leq_m \overline{HALT}_TM$ (1), and $\overline{A}_TM$ is unrecognizable (Corollary 4.23)
Exercise 8 (Level 3)

(Exercise 4.2 in the book.)

\[ EQ_{DFAREX} \{ \langle D, R \rangle \mid D \text{ is a DFA} \land R \text{ is a regex} \land L(D) = L(R) \} \]
Exercise 8 (Level 3)

(Exercise 4.2 in the book.)

\[ EQ_{DFAREX} \{ \langle D, R \rangle \mid D \text{ is a DFA} \land R \text{ is a regex} \land L(D) = L(R) \} \]

Let \( r2n \) be the function that converts a regular expression into an NFA and \( n2d \) be the function that converts an NFA into a DFA.

1. \( EQ_{DFAREX} \leq_m EQ_{DFA} \) with \( F(\langle D, R \rangle) = \langle D, n2d(r2n(R)) \rangle \).
   - Unfold \( \leq_m \). Goal: \( \langle D, R \rangle \in EQ_{DFAREX} \iff F(\langle D, R \rangle) \in EQ_{DFA} \)
   - Unfold \( EQ_{DFAREX}, EQ_{DFA}, \) and \( F \). Goal: \( L(D) = L(R) \iff L(D) = n2d(r2n(R)) \)
   - Rewrite goal with \( \forall N, L(n2d(N)) = L(N) \) and \( \forall R, r2n(R) = L(R) \). Goal: \( L(D) = L(R) \iff L(D) = L(R) \)
     - Proof: trivial, since \( \forall P, P \iff P \).

2. \( EQ_{DFAREX} \) is decidable, by Theorem 5.22, (1) \( EQ_{DFAREX} \leq_m EQ_{DFA} \), and \( EQ_{DFA} \) decidable (Theorem 4.5).

The proof has two main parts: 1) showing that the given language map-reduces to a decidable language and 2) use Theorem 5.22 to conclude.
Exercise 8 (Level 3)

Continuation...

- The proof has two main parts: 1) showing that the given language map-reduces to a decidable language and 2) use Theorem 5.22 to conclude.
- Whenever you say that $A \leq_m B$ be clear about which function reduces $A$ to $B$.

More examples

- See HW7
Exercise 9 (Level 3)

Hint...

Combine Lemma R.1, R.2, R.3, Exercise 5.6, Exercise 5.22, and decidability, recognizability to relate the recognizability/decidability between mapping-reducible languages.
Thank you!